HEAT

McDonald, David; He, Shan

DOI:
10.1007/978-3-030-62362-3_4

HEAT: Hyperbolic Embedding of Attributed Networks

David McDonald[1][0000−0002−0540−8254] and Shan He[1][0000−0003−1694−1465]

School of Computer Science, University of Birmingham, Edgbaston, Birmingham, B15 2TT, United Kingdom
office@cs.bham.ac.uk
https://www.cs.bham.ac.uk {dxm237,s.he}@cs.bham.ac.uk

Abstract. Finding a low dimensional representation of hierarchical, structured data described by a network remains a challenging problem in the machine learning community. An emerging approach is embedding *networks* into hyperbolic space because it can naturally represent a network's hierarchical structure. However, existing hyperbolic embedding approaches cannot deal with attributed networks, in which nodes are annotated with additional attributes. These attributes might provide additional proximity information to constrain the representations of the nodes, which is important to learn high quality hyperbolic embeddings. To overcome this gap we propose HEAT (Hyperbolic Embedding of Attributed Networks). HEAT first extracts training samples from the original graph capturing both topological and attribute similarity and then learns a hyperboloid embedding using full Riemannian Stochastic Gradient Descent. We show that HEAT can outperform other network embedding algorithms on several downstream tasks. As a general embedding method, HEAT opens the door to hyperbolic manifold learning on a wide range of attributed and unattributed networks.

Keywords: Network embedding · Hyperbolic embedding · Random walk.

1 Introduction

The success of machine learning algorithms often depends upon data representation [1]. Unsupervised representation learning – learning alternative (low dimensional) representations of data – has become common for processing information on non-Euclidean domains, such as complex networks. Prediction over nodes and edges requires careful feature engineering [2] and representation learning leads to the extraction of features from a graph that are most useful for downstream tasks, without careful design or a-priori knowledge.

An emerging representation learning approach for complex networks is hyperbolic embedding. This approach is based on compelling evidence that the underlying metric space of many complex networks is hyperbolic [3]. A hyperbolic space can be interpreted as a continuous representation of a discrete tree structure that captures the hierarchical organisation of elements within a complex

system [3]. Furthermore, hyperbolic metric spaces have been shown to explain other characteristic typical to complex networks, characteristics such as clustering [3] and the "small world" phenomenon [4]. Hyperbolic spaces therefore offer a natural continuous representations of hierarchical complex networks [3].

However, existing hyperbolic embedding approaches cannot deal with attributed networks, of which nodes (entities) are richly annotated with attributes [5]. For example, a paper within a citation network may be annotated with the presence of keywords, and the people in a social network might have additional information such as interests, hobbies, and place of work. These attributes might provide additional proximity information to constrain the representations of the nodes. Therefore, incorporating node attributes can improve the quality of the final embedding, with respect to many different downstream tasks [5].

This paper proposes the first hyperbolic embedding method for attributed networks called HEAT. The intuition behind HEAT is to extract training samples from the original graph, which can capture both topological and attribute similarities, and then learn a hyperbolic embedding based on these samples. To extract training samples, a novel random walk algorithm with a teleport procedure is developed. The purpose of this walk is to capture phantom links between nodes that do not necessarily share a topological link, but have highly similar attributes. To learn the embeddings from these extracted samples, HEAT employs a novel learning objective that is optimized using full Riemannian stochastic gradient descent in hyperbolic space.

Thorough experimentation shows that HEAT can achieve better performance on several downstream tasks compared with several state-of-the-art embedding algorithms. As a general framework, HEAT can embed both unattributed and attributed networks with continuous and discrete attributes, which opens the door to hyperbolic manifold learning for a wide range of complex networks.

1.1 Related Work

Recently, graph convolution has been generalised to hyperbolic space to allow for non-Euclidean representation learning on attributed networks [6]. The algorithm in [7] embeds networks to the Poincaré ball using retraction updates to optimize an objective that aims to maximize the likelihood of observing true node pairs versus arbitrary pairs of nodes in the network. Also, trees can be embedded in hyperbolic space without distortion [8] and so, some works by embed general graphs to trees and then compute an exact distortion-free embedding of the resulting tree [9].

For attributed network embedding in Euclidean space, several algorithms have been proposed. Text-assisted Deepwalk (TADW) [10] generalises Deepwalk [11] to nodes with text attributes. By generalising convolution from regular pixel lattices to arbitrary graphs, it is possible embed and classify entire graphs [12]. Furthermore, the popular Graph Convolutional Network (GCN) extend this approach to simplify graph convolution in the semi-supervised setting [13]. GraphSAGE introduces an inductive framework for online learning of node embeddings capable of generalising to unseen nodes [5].

2 Hyperboloid Model of Hyperbolic Geometry

Due to the fundamental difficulties in representing spaces of constant negative curvature as subsets of Euclidean spaces, there are not one but many equivalent models of hyperbolic spaces [14]. The models are equivalent because all models of hyperbolic geometry can be freely mapped to each other by a distance preserving mapping called an *isometry*. Each model emphasizes different aspects of hyperbolic geometry, but no model simultaneously represents all of its properties.

For HEAT, the hyperboloid model is selected for the simple form of Riemannian operations. The main advantage of this is that the simple forms allow for the inexpensive computation of exact gradients and, therefore, HEAT is optimized using full Riemannian Stochastic Gradient Descent (RSGD) [15], rather than approximating gradient descent with retraction updates [7]. Ultimately, this has the advantage of faster convergence [15]. Unlike disk models, that sit in an ambient Euclidean space of dimension n, the hyperboloid model of n-dimensional hyperbolic geometry sits in $n+1$-dimensional Minkowski space-time. Minkowski space-time is denoted $\mathbb{R}^{n:1}$, and a point $\mathbf{x} \in \mathbb{R}^{n:1}$ has spacial coordinates $\mathbf{x}^i$ for $i = 1, 2, ..., n$ and time co-ordinate $\mathbf{x}^{n+1}$.

HEAT requires the *Minkowski bilinear form* for both its learning objective as well as for parameter optimization. The Minkowski bilinear form is defined as $\langle \mathbf{u}, \mathbf{v} \rangle_{\mathbb{R}^{n:1}} = \sum_{i=1}^{n} \mathbf{u}^i \mathbf{v}^i - \psi^2 \mathbf{u}^{n+1} \mathbf{v}^{n+1}$ where ψ is the speed of information flow in our system (here set to 1 for simplified calculations). This bilinear form is an inner product and allows the computation of the Minkowski norm: $||\mathbf{u}||_{\mathbb{R}^{n:1}} := \sqrt{\langle \mathbf{u}, \mathbf{u} \rangle_{\mathbb{R}^{n:1}}}$. Using the bilinear form, the hyperboloid is defined as $\mathbb{H}^n = \{\mathbf{u} \in \mathbb{R}^{n:1} \mid \langle \mathbf{u}, \mathbf{u} \rangle_{\mathbb{R}^{n:1}} = -1, \mathbf{u}^{n+1} > 0\}$. The first condition defines a hyperbola of two sheets, and the second one selects the top sheet.

In addition, the bilinear form is required to define the distance between two points on the hyperboloid, which is incorporated as part of HEAT's training objective function. The distance between two points $\mathbf{u}, \mathbf{v} \in \mathbb{H}^n$ is given by the length of the geodesic between them $D_{\mathbb{H}^n}(\mathbf{u}, \mathbf{v}) = \operatorname{arccosh}(\gamma)$ where $\gamma = -\langle \mathbf{u}, \mathbf{v} \rangle_{\mathbb{R}^{n:1}}$ [16].

To update parameters, HEAT performs full RSGD. This requires defining the tangent space of a point $\mathbf{u} \in \mathbb{H}^n$, that is denoted $T_{\mathbf{u}}\mathbb{H}^n$. It is the collection of all points in $\mathbb{R}^{n:1}$ that are orthogonal to $\mathbf{u}$, and is defined as $T_{\mathbf{u}}\mathbb{H}^n = \{\mathbf{x} \in \mathbb{R}^{n:1} \mid \langle \mathbf{u}, \mathbf{x} \rangle_{\mathbb{R}^{n:1}} = 0\}$. It can be shown that $\langle \mathbf{x}, \mathbf{x} \rangle_{\mathbb{R}^{n:1}} > 0 \ \forall \mathbf{x} \in T_{\mathbf{u}}\mathbb{H}^n \ \forall \mathbf{u} \in \mathbb{H}^n$, and so the tangent space of every point the hyperboloid is positive-definite, and so $\mathbb{H}^n$ is a Riemannian manifold [16].

3 Hyperbolic Embedding of Attributed Networks

3.1 Problem Definition

We consider a network of N nodes given by the set V with $|V| = N$. We use E to denote the set of all interactions between the nodes in our network. $E = \{(u, v)\} \subseteq V \times V$. We use the matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$ to encode the weights of these interactions, where $\mathbf{W}_{uv}$ is the weight of the interaction between node u and

node v. We have that $\mathbf{W}_{uv} \neq 0 \iff (u, v) \in E$. If the network is unweighted then $\mathbf{W}_{uv} = 1$ for all $(u, v) \in E$. Furthermore, the matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$ describes the attributes of each node in the network. These attributes may be discrete or continuous. Edge attributes could be handled by transforming them into node attributes shared by both nodes connected by the edge. We consider the problem of representing a graph given as $\mathbb{G} = (V, E, \mathbf{W}, \mathbf{X})$ as set of low-dimensional vectors in the n-dimensional hyperboloid $\{\mathbf{x}_v \in \mathbb{H}^n \mid v \in V\}$, with $n \ll N$. The described problem is unsupervised.

3.2 HEAT Overview

Our proposed HEAT consists of two main components:

1. A novel network sampling algorithm based on random walks to extract samples than can capture both topological and attribute similarity.
2. A novel learning algorithm that can learn hyperbolic embeddings from training samples using Riemannian stochastic gradient descent (RSGD) in hyperbolic space.

3.3 Sample the Network using Random Walks with Jump

We propose a novel random-walk procedure to obtain training samples that capture both topological and attribute similarity. Random walks have been proposed in the past as a robust sampling method of elements from structured data, such as graphs, since they provide an efficient, flexible and parallelizable sampling method [11]. For every node in the network, several 'walks' with a fixed length l are performed [2]. We note that random walks traditionally take into account only first-order topological similarity, that is: nodes are similar if they are connected in the network. However, additional topological similarity can be considered. For example, second-order similarity between nodes (that is: the similarity of neighbourhoods) could be incorporated into the topological similarity matrix using a weighted sum [17]. We leave this as future work.

We propose that, in addition to standard random walks which capture topological similarity, we use attribute similarity to 'jump' the random walker to the nodes with similar attributes. This approach is inspired by PageRank [18], where the random walk jumps from a node to any other node in the web graph is the current location of the walk has no out-links, and is similar to RoSANE [19]. To this end, we define the attribute similarity $\mathbf{Y}$ as cosine similarity of the attribute vectors of the nodes. We assign a value of 0 similarity to any negative similarity values. We select cosine similarity as it can readily handle high dimensional data well without making a strong assumption about the data. We propose HEAT as a general framework and, so, can change the cosine similarity to a more sophisticated and problem-dependant measure of pairwise node attribute similarity.

To define the probability of moving from a node to another based on both topological and attribute similarity, we then additionally define $\bar{\mathbf{W}}$ and $\bar{\mathbf{Y}}$ to be the row-normalized versions of the weight matrix $\mathbf{W}$ and attribute similarity matrix $\mathbf{Y}$ respectively. Each row in $\bar{\mathbf{W}}$ and $\bar{\mathbf{Y}}$ describes a discrete probability distribution corresponding to the likelihood of jumping from one node to the next according to either topological similarity or attribute similarity.

To control the trade-off between topology and attributes, we define the hyper-parameter $0 \leq \alpha \leq 1$. Formally, we use i to denote the ith node in the walk ($x_0 = s$), and for each step $i = 1, 2, ..., l$ in the walk, we sample $\pi_i \sim U(0, 1)$ and determine the ith node as follows: if $\pi_i < \alpha$, then $P(x_i = v \mid x_{i-1} = u) = \hat{\mathbf{Y}}_{uv}$, else $P(x_i = v \mid x_{i-1} = u) = \hat{\mathbf{W}}_{uv}$.

We follow previous works, and consider nodes that appear within a maximum distance of each other in the same walk to be "context pairs" [2,11]. We call this maximum distance the "context-size" and it is a hyper-parameter that controls the size of a local neighbourhood of a node. Previous works show that increasing context size typically improves performance, at some computational cost [2]. All of the source-context pairs are added into a set $\mathcal{D}$.

3.4 Hyperboloid Embedding Learning

For the hyperboloid embedding learning procedure of HEAT, we aim to maximise the probability of observing all of the pairs in $\mathcal{D}$ in the low-dimensional embedded space. We define the probability of two nodes sharing a connection to be a function of their distance in the embedding space. This is motivated by the intuition of network embedding that nodes separated by a small distances share a high degree of similarity and should, therefore share a high probability of connection, and nodes very far apart in the embedding space should share a low probability of connection. This principle forms the basis of an objective function that is optimized by HEAT to learn node embeddings.

We make the common simplifying assumption that the distance between a source node and neighbourhood is symmetric (ie: $P((u, v)) = P((v, u))$ for all $u, v \in V$) [2]. To this end, we define the symmetric function $\hat{P}((u, v)) := -D^2_{\mathbb{H}^n}(\mathbf{u}, \mathbf{v})$ to be the unnormalized probability of observing a link between source node u and context node v, where $\mathbf{u}$ and $\mathbf{v}$ are their respective hyperbolic positions. We square the distance because this leads to stable gradients [9]. We normalise the probability using: $P((u, v)) := \exp(\hat{P}((u, v)))/Z(u)$, where $Z(u) := \sum_{v' \in V} \exp(\hat{P}((u, v')))$.

However, computing the gradient of the partition function $Z(u)$ involves a summation over all nodes $v \in V$, which for large networks, is prohibitively computationally expensive [2]. Following previous works, we overcome this limitation through *negative sampling*. We define the set of negative samples for u, as the set of v for we we observe no relation with u: $\text{Neg}(u) := \{v \in V \mid (u, v) \notin \mathcal{D}\}$. We further define $\text{Neg}_K(u, v) := \{x_i \sim^{P_n} \text{Neg}(u) \mid i = 1, 2, ..., K\} \cup \{v\}$ to be a random sample with replacement of size K from the set of negative samples of u, according to a noise distribution P_n including v. Following [2], we set $P_n = U^{\frac{3}{4}}$, the unigram distribution raised to the $3/4$ power.

We aim to represent the obtained distribution of pairs in a low-dimensional hyperbolic space. To this end, we formulate a loss function L that encourages maximising the probability of observing all positive sample pairs $P(v \mid u)$ for all $(u, v) \in \mathcal{D}$ and minimising the probability of observing all other pairs. To this end, we define the loss function L for an embedding $\Theta = \{\mathbf{u} \in \mathbb{H}^n \mid u \in V\}$ to be the the mean of negative log-likelihood of observing all the source-context pairs in $\mathcal{D}$, against the negative sample noise:

$$L(\Theta) = -\frac{1}{|\mathcal{D}|} \sum_{(u,v)\in\mathcal{D}} \log\left[\frac{\exp\left(-D^2_{\mathbb{H}^n}(\mathbf{u},\mathbf{v})\right)}{\sum_{v'\in \mathrm{Neg}_K(u,v)} \exp\left(-D^2_{\mathbb{H}^n}(\mathbf{u},\mathbf{v}')\right)}\right] \tag{1}$$

The numerator of eq. (1), $\exp\left(-D^2_{\mathbb{H}^n}(\mathbf{u},\mathbf{v})\right)$, is concerned with the hyperbolic distance between nodes u and v in the positive sample set $\mathcal{D}$. Minimising L involves minimising the distance between $\mathbf{u}$ and $\mathbf{v}$ in the embedding space. The denominator is a sum over all v' in a given sample of size m of the negative samples for node u. Minimising L involves maximising this term, thereby pushing $\mathbf{u}$ and $\mathbf{v}'$ far apart in the embedding space. Overall, we observe that minimising L involves maximising $P((u,v))$ for all $(u,v) \in \mathcal{D}$ as required. This encourages source-context pairs to be close together in the embedding space, and u to be embedded far from the noise nodes v' [7].

3.5 Optimization

Since we use hyperboloid model, unlike some previous works [7, 9] that use the *Poincaré ball* model and approximate gradients with retraction updates, we are able to use full Riemannian optimization and so our gradient computation is exact and possesses a simple form [20]. We follow a three step procedure in to compute gradients and then update hyperbolic coordinates [20].

To compute the gradient of L for vector $\mathbf{u} \in \mathbb{H}^n$ with respect to the hyperboloid $\nabla^{\mathbb{H}^n}_{\mathbf{u}} L$, we first compute the gradient with respect to the Minkowski ambient space $\mathbb{R}^{n:1}$ as

$$\nabla^{\mathbb{R}^{n:1}}_{\mathbf{u}} L = \left(\frac{\partial L}{\partial u^1}\bigg|_{\mathbf{u}}, ..., \frac{\partial L}{\partial u^n}\bigg|_{\mathbf{u}}, -\frac{\partial L}{\partial u^{n+1}}\bigg|_{\mathbf{u}}\right) \tag{2}$$

Let $o_{uv} := -D^2_{\mathbb{H}^n}(\mathbf{u},\mathbf{v})$ and $\mathrm{Neg}_K(u) := \bigcup_{\{v|(u,v)\in\mathcal{D}\}} \mathrm{Neg}_K(u,v)$. Then

$$\nabla^{\mathbb{R}^{n:1}}_{\mathbf{u}} L = \frac{1}{|\mathcal{D}|} \sum_{v\in \mathrm{Neg}_K(u)} (\delta_{vv'} - P((u,v))) \cdot \nabla^{\mathbb{R}^{n:1}}_{\mathbf{u}} o_{uv'} \tag{3}$$

and

$$\nabla^{\mathbb{R}^{n:1}}_{\mathbf{u}} o_{uv} = 2 \cdot \frac{\mathrm{arccosh}\left(-\langle \mathbf{u},\mathbf{v}\rangle_{\mathbb{R}^{n:1}}\right)}{\sqrt{\langle \mathbf{u},\mathbf{v}\rangle^2_{\mathbb{R}^{n:1}} - 1}} \cdot \mathbf{v} \tag{4}$$

where $\delta_{vv'}$ is the Kronecker delta function. We then use the vector projection formula to compute the projection of the ambient gradient to its component in the tangent space:

$$\nabla^{\mathbb{H}^n}_{\mathbf{u}} L = \nabla^{\mathbb{R}^{n:1}}_{\mathbf{u}} L + \langle \mathbf{u}, \nabla^{\mathbb{R}^{n:1}}_{\mathbf{u}} L\rangle_{\mathbb{R}^{n:1}} \cdot \mathbf{u} \tag{5}$$

Having computed the gradient component in the tangent space of $\mathbf{u}$, we define the exponential map to take a vector $\mathbf{x} \in T_{\mathbf{u}}\mathbb{H}^n$ to its corresponding point on the hyperboloid:

$$\mathrm{Exp}_{\mathbf{u}}(\mathbf{x}) = \cosh(r) \cdot \mathbf{u} + \sinh(r) \cdot \frac{\mathbf{x}}{r} \tag{6}$$

Where $r = ||\mathbf{x}||_{\mathbb{R}^{n:1}} = \sqrt{\langle \mathbf{x},\mathbf{x}\rangle_{\mathbb{R}^{n:1}}}$ denotes the Minkowski norm of $\mathbf{x}$.

Altogether, we have that the three-step procedure for computing the new position of $\mathbf{u}$, with learning rate η is:

1. Calculate ambient gradient $\nabla^{\mathbb{R}^{n:1}}_{\mathbf{u}} L$ (eqs. (3) and (4)),

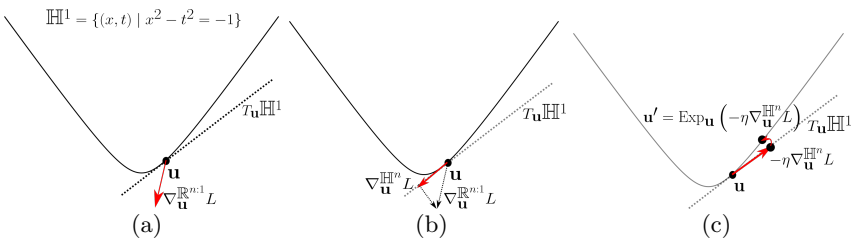


Fig. 1: Exponential map on $\mathbb{H}^n$. (a) provides a representation of $\mathbb{H}^1$ a one dimensional manifold in two dimensional Minkowski space $\mathbb{R}^{1:1}$. One point on $\mathbb{H}^1$ is highlighted: $\mathbf{u}$. The red arrow is an example $\nabla^{\mathbb{R}^{n:1}}_{\mathbf{x}_u} L$ vector. Finally, $T_{\mathbf{u}}\mathbb{H}^n$ is given by the dotted black line. (b) highlights the component of $\nabla^{\mathbb{R}^{n:1}}_{\mathbf{x}_u} L$ lying on $T_{\mathbf{u}}\mathbb{H}^n$. Finally, (c) plots the mapping from $-\eta \nabla^{\mathbb{H}^n}_{\mathbf{u}} L$ back to the Hyperboloid using the exponential map $\mathrm{Exp}_{\mathbf{u}}$.

Table 1: Network statistics. *Key:* N is the number of nodes, $|E|$ is the number of edges, d is the dimension of node features, y is the number of classes.

Network	N	$\lvert E\rvert$	d	y
Cora_ML [21]	2995	8416	2879	7
Citeseer [21]	4230	5358	2701	6
Pubmed [21]	18230	79612	500	3
PPI (LCC) [5]	3480	54806	50	121
MIT [19]	6402	251230	2804	32

2. Project $\nabla^{\mathbb{R}^{n:1}}_{\mathbf{u}} L$ to tangent $\nabla^{\mathbb{H}^n}_{\mathbf{u}} L$ (eq. (5)),
3. Set $\mathbf{u} = \mathrm{Exp}_{\mathbf{u}}\left(-\eta \nabla^{\mathbb{H}^n}_{\mathbf{u}} L\right)$ (eq. (6)).

Figure 1 provides an example of this procedure operating on $\mathbb{H}^1$.

3.6 Setting α

For practical applications, we suggest the following approach to set thew value of the hyper-parameter α that controls the trade-off between topology and similarity in the sampling process: Randomly sample a proportion of edges from the network and remove them. Next, select an equal number of 'non-edge' node pairs $(u, v) \in V \times V \setminus E$. These two edge sets will form a validation set. Then perform HEAT to generate hyperboloid embeddings for a range of values of α. Removed edges can be ranked against the sampled non-edges and a global ranking measure, such as AUROC or AP could be used to select a value for α. This procedure is performed in section 4.4, where we evaluate link prediction. As we show in section 4.5, HEAT is robust to the setting of α for three common downstream tasks.

4 Experimental Validation

4.1 Datasets

We evaluate HEAT on three citation networks [21], one PPI network [5], and one social network for MIT university [19]. Features for all networks were scaled

Table 2: Description of benchmark algorithms.

Algorithm	Description
DEEPWALK [11]	One of the most successful representation learning approaches based on random walks. Considers only structural information.
ATTRPURE [19]	Performs SVD on a devised attribute similarity matrix.
TADW [10]	Uses matrix factorisation to jointly model attribute and structural information.
AANE [22]	A distributed alternative to TADW.
SAGEGCN [13]	Graph convolutional network (GCN) originally designed for semi-supervised learning but adapted in [5] for unsupervised learning.
N&K [7]	Hyperbolic embedding approach based only on structural information.

to have a mean of 0 and a standard deviation of 1. Table 1 shows the statistics of these five networks.

4.2 Benchmark Algorithms and Settings

Table 2 details all of the benchmark algorithms. For all benchmark methods we adopt the original source code. We train all methods in an unsupervised manner.

4.3 Network Reconstruction & Link Prediction

Following common practice, we use network reconstruction to evaluate the capacity of the learned embeddings to reflect the original data [7]. After training our model to convergence upon the complete information, we compute distances in the embedding space between all pairs of nodes according to both models.

To evaluate the link prediction ability of the learned embeddings, we randomly select 10% of the edges in the network and remove them [7]. We then randomly select also an equal number of true non-edges in the network. An embedding is learned for each incomplete network and pairs of nodes are ranked by distance.

Table 3 provides a summary of the network reconstruction and link prediction results. For reconstruction, we observe that HEAT has a high capacity for learning network structure, even at low dimensions. Further, by incorporating attributes, performance increased further on two out of the five networks studied. The link prediction results demonstrate that HEAT is capable of highly competitive link prediction ability with and without attributes. We see that the inclusion of attributes improves performance on three out of five networks and suggest that this is because of the high level of homophily in citation networks.

4.4 Node Classification

To evaluate node classification, we learn an embedding, using complete topological and attribute information, in an unsupervised manner. We then use an out-of-the-box Support Vector Classifier (SVC) to evaluate the separation of classes in the embedding space For hyperbolic embeddings (HEAT and N&K), we first project to the Klein model of hyperbolic space, which preserves straight lines [14]. For the PPI network, each protein has multiple labels from the Gene Ontology [5]. To evaluate our model in this multi-label case, we adopt a one-vs-all setting, where we train a separate classifier for each class.

Table 3: Summary of network reconstruction and link prediction results for embedding dimension 10. We present AUROC and AP scores, mean average precision (mAP) and precisions at k (p@k) for $k \in \{1, 3, 5, 10\}$ to 3 decimal places and rank to 1 decimal place. Rank is the average position that a true edge appears in the list of false edges ranked by distance. For mAP, we rank distances with respect to each node in the network and compute a separate precision score for each node, then report the mean of these precisions. For p@k, report the number of the k closest nodes that are true neighbours. All scores are averaged over 30 random starting seeds. $\text{HEAT}_{\alpha=0.2}$ considers node attributes, whereas $\text{HEAT}_{\alpha=0.0}$ does not. Bold text indicates the best score.

	Reconstruction								Link Prediction			
Cora_ML												
	Rank	AUROC	AP	mAP	p@1	p@3	p@5	p@10	Rank	AUROC	AP	mAP
N&K	209.8	0.987	0.986	0.674	0.791	0.736	0.723	0.713	96.6	0.922	0.935	0.248
AANE	3945.7	0.758	0.775	0.145	0.182	0.192	0.206	0.238	315.3	0.743	0.761	0.098
TADW	539.9	0.967	0.959	0.401	0.458	0.425	0.438	0.444	72.8	0.941	0.933	0.180
ATTRPURE	4768.0	0.708	0.735	0.124	0.156	0.164	0.174	0.203	356.5	0.710	0.736	0.093
DEEPWALK	185.6	0.989	0.986	0.712	0.757	0.723	0.722	0.720	178.9	0.855	0.896	0.224
SAGEGCN	913.9	0.944	0.935	0.289	0.304	0.341	0.363	0.392	160.5	0.870	0.873	0.114
$\text{HEAT}_{\alpha=0.00}$	**40.9**	**0.998**	**0.997**	**0.853**	0.874	**0.869**	**0.858**	**0.850**	131.8	0.893	0.928	0.276
$\text{HEAT}_{\alpha=0.20}$	58.6	0.996	0.995	0.838	**0.895**	0.838	0.823	0.813	**49.0**	**0.961**	**0.963**	**0.288**
Citeseer												
N&K	67.5	0.994	0.994	0.799	0.774	0.772	0.800	0.837	145.5	0.820	0.853	0.204
AANE	3741.9	0.650	0.645	0.097	0.088	0.112	0.132	0.196	284.8	0.646	0.643	0.086
TADW	297.1	0.972	0.964	0.376	0.314	0.352	0.399	0.469	55.9	0.931	0.917	0.149
ATTRPURE	3922.2	0.633	0.630	0.089	0.083	0.102	0.126	0.189	297.1	0.630	0.630	0.083
DEEPWALK	23.5	0.998	0.997	0.798	0.721	0.805	0.853	0.899	259.9	0.677	0.775	0.213
SAGEGCN	618.9	0.942	0.939	0.216	0.160	0.291	0.364	0.470	138.1	0.829	0.848	0.132
$\text{HEAT}_{\alpha=0.00}$	9.6	0.999	0.999	0.830	0.740	**0.898**	**0.932**	**0.967**	16.5	0.981	0.979	0.791
$\text{HEAT}_{\alpha=0.20}$	**7.6**	**0.999**	**0.999**	**0.926**	**0.899**	0.894	0.908	0.933	**6.3**	**0.993**	**0.994**	**0.810**
Pubmed												
N&K	451.5	0.995	0.994	0.737	0.743	0.803	0.835	0.835	801.7	0.880	0.900	0.210
AANE	19746.8	0.777	0.784	0.104	0.106	0.187	0.219	0.253	1503.8	0.774	0.782	0.088
TADW	2155.6	0.976	0.973	0.514	0.502	0.550	0.600	0.617	465.5	0.930	0.923	0.157
ATTRPURE	25982.8	0.707	0.707	0.097	0.098	0.175	0.207	0.241	1954.5	0.706	0.706	0.082
DEEPWALK	565.1	0.994	0.993	0.816	0.794	0.808	0.845	0.856	1742.8	0.738	0.833	0.203
SAGEGCN	4118.9	0.954	0.945	0.261	0.215	0.341	0.405	0.467	635.5	0.905	0.901	0.114
$\text{HEAT}_{\alpha=0.00}$	**115.8**	**0.999**	0.998	0.823	0.753	0.863	**0.896**	**0.905**	463.6	0.930	0.931	0.202
$\text{HEAT}_{\alpha=0.20}$	161.8	0.998	**0.998**	**0.908**	**0.913**	**0.876**	0.892	0.893	**322.8**	**0.952**	**0.954**	**0.256**
PPI												
N&K	6757.7	0.938	0.940	0.421	**0.801**	**0.699**	**0.664**	0.640	722.1	0.912	0.918	0.185
AANE	50719.7	0.537	0.588	0.089	0.470	0.251	0.201	0.168	3822.2	0.535	0.582	0.070
TADW	23408.8	0.786	0.767	0.142	0.502	0.298	0.258	0.242	2237.5	0.728	0.722	0.080
ATTRPURE	51304.6	0.511	0.512	0.038	0.164	0.084	0.065	0.051	3854.5	0.511	0.511	0.018
DEEPWALK	9962.9	0.909	0.903	0.388	0.721	0.582	0.543	0.523	1066.4	0.870	0.873	0.144
SAGEGCN	44688.5	0.592	0.607	0.095	0.455	0.218	0.169	0.137	3622.2	0.560	0.574	0.067
$\text{HEAT}_{\alpha=0.00}$	**4926.0**	**0.955**	**0.952**	**0.468**	0.782	0.696	0.664	**0.643**	**431.6**	**0.948**	**0.947**	**0.255**
$\text{HEAT}_{\alpha=0.20}$	5628.5	0.949	0.946	0.457	0.786	0.667	0.627	0.604	493.0	0.940	0.940	0.241
MIT												
N&K	32506.7	0.927	0.930	0.565	**1.000**	0.884	0.855	0.827	2727.1	0.918	0.922	0.256
AANE	157174.0	0.647	0.639	0.189	**1.000**	0.528	0.414	0.321	11816.8	0.646	0.638	0.098
TADW	79981.9	0.820	0.814	0.397	**1.000**	0.760	0.700	0.639	6133.4	0.816	0.815	0.180
ATTRPURE	171241.0	0.615	0.617	0.184	0.993	0.543	0.428	0.323	12823.6	0.616	0.617	0.098
DEEPWALK	33037.6	0.926	0.919	0.578	**1.000**	0.857	0.819	0.782	2834.4	0.915	0.909	0.241
SAGEGCN	89637.4	0.798	0.797	0.380	**1.000**	0.670	0.599	0.545	7142.0	0.786	0.784	0.153
$\text{HEAT}_{\alpha=0.00}$	**24776.7**	**0.944**	**0.940**	**0.639**	**1.000**	**0.902**	**0.877**	**0.850**	**2246.9**	**0.933**	**0.930**	**0.279**
$\text{HEAT}_{\alpha=0.20}$	28621.0	0.936	0.934	0.633	**1.000**	0.901	0.870	0.839	2510.4	0.925	0.925	0.273

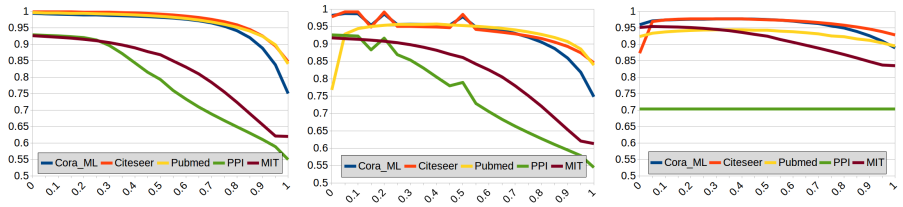
Table 4: Summary of node classification results for embedding dimension 10. All measures are micro-averaged. $\text{HEAT}_{\alpha=0.2}$ considers node attributes, whereas $\text{HEAT}_{\alpha=0.0}$ does not. Bold indicates best performance. Mean Rank on All Datasets is the average position of an algorithm in a ranked list of performance.

	Cora_ML				PPI			
	F1	Precision	Recall	AUROC	F1	Precision	Recall	AUROC
N&K	0.761	0.827	0.704	0.953	0.387	**0.695**	0.268	0.704
AANE	0.706	0.814	0.624	0.944	0.395	0.688	0.277	0.705
TADW	0.837	0.869	0.808	**0.983**	0.393	0.688	0.275	0.705
ATTRPURE	0.675	0.782	0.594	0.937	**0.398**	0.688	**0.281**	**0.705**
DEEPWALK	**0.855**	**0.886**	**0.827**	0.977	0.387	0.694	0.269	0.704
SAGEGCN	0.679	0.758	0.616	0.933	0.388	0.694	0.269	0.704
$\text{HEAT}_{\alpha=0.0}$	0.804	0.858	0.756	0.965	0.388	0.694	0.269	0.704
$\text{HEAT}_{\alpha=0.2}$	0.849	0.884	0.817	0.980	0.388	0.694	0.269	0.704
	Citeseer				MIT			
	F1	Precision	Recall	AUROC	F1	Precision	Recall	AUROC
N&K	0.516	0.856	0.370	0.861	0.508	**0.858**	0.361	0.937
AANE	0.609	0.829	0.482	0.897	0.036	0.414	0.019	0.866
TADW	0.878	0.899	0.857	0.980	0.544	0.739	0.430	0.949
ATTRPURE	0.597	0.837	0.465	0.892	0.056	0.409	0.030	0.869
DEEPWALK	**0.927**	**0.939**	**0.916**	**0.987**	**0.661**	0.841	**0.544**	**0.965**
SAGEGCN	0.466	0.727	0.347	0.843	0.480	0.832	0.337	0.929
$\text{HEAT}_{\alpha=0.0}$	0.594	0.857	0.455	0.879	0.634	0.849	0.507	0.959
$\text{HEAT}_{\alpha=0.2}$	0.887	0.914	0.861	0.981	0.637	0.819	0.522	0.960
	Pubmed				Mean Rank on All Datasets			
	F1	Precision	Recall	AUROC	F1	Precision	Recall	AUROC
N&K	0.797	0.815	0.780	0.933	5.4	3	5.6	5.6
AANE	**0.830**	**0.840**	**0.819**	**0.947**	4.2	5.8	4.2	4.2
TADW	0.778	0.797	0.761	0.921	3.8	4.8	3.6	3.2
ATTRPURE	0.781	0.806	0.758	0.912	5.2	6.6	5.4	5.2
DEEPWALK	0.765	0.792	0.740	0.907	3.8	3.2	3.6	3.2
SAGEGCN	0.768	0.791	0.746	0.906	6.8	6.6	6.8	7.4
$\text{HEAT}_{\alpha=0.0}$	0.807	0.823	0.791	0.933	4	3	4	4
$\text{HEAT}_{\alpha=0.2}$	0.797	0.815	0.780	0.933	**2.6**	**2.8**	**2.6**	**3**

Table 4 proves a summary of the node classification results. While HEAT never ranked first for any network, it consistently ranked highly on all networks unlike all other benchmark algorithms – with $\text{HEAT}_{\alpha=0.2}$ the best overall rank averaged across all the datasets. Further, we observe that when considering node attributes, i.e., $\alpha = 0.2$, HEAT outperformed the other hyperbolic embedding algorithm N&K on all networks. We also note that, even without node attributes, HEAT obtained better results than N&K on all networks. Comparing with the Euclidean benchmark algorithms, we observe competitive results – especially when incorporating attributes.

4.5 Sensitivity of Control Parameters

We carried out preliminary experiments to evaluate HEAT's robustness to the setting of the control parameters. Our results indicated that the most sensitive parameter is α, which controls the trade off between considering topology and attributes. We run HEAT over a range of values $\alpha \in [0, 1]$ in steps of 0.05.



(a) Network reconstruction (b) Link prediction (c) Node classification

Fig. 2: The effect of the setting of α (x-axis) on the AUROC scores (y-axis) on three downstream machine learning tasks: (a) network reconstruction, (b) link prediction, and (c) node classification.

Figure 2 plots AUROC scores for network reconstruction, link prediction and node classification obtained from a 5 dimensional embedding. From the three plots, we observe that the performance of HEAT on the three tasks on all of the networks is robust to a wide range of values of α (especially $\alpha \in [0, 0.5]$).

5 Conclusion

This paper presents HEAT to fill the gap of embedding attributed networks in hyperbolic space. We have designed a random walk algorithm to obtain the training samples that capture both network topological and attribute similarity. We have also derived an algorithm that learns hyperboloid embeddings from the training samples. Our results on five benchmark attributed networks show that, by including attributes, HEAT can improve the quality of a learned hyperbolic embedding in a number of downstream machine learning tasks. We find that, in general, HEAT does not perform as well in the node classification task compared with the Euclidean benchmark algorithms (table 4) than in the network reconstruction and link prediction tasks (table 3): while it is the most consistent across all datasets, it does not rank first on any particular dataset as it does for reconstruction and link prediction. This could be attributed to the SVC learning sub-optimal decision boundaries, since it is using a Euclidean optimization procedure. Logistic regression has been generalized to the Poincaré ball [23], and this may provide superior results. However, we leave this as future work.

Overall, HEAT provides a general hyperbolic embedding method for both unattributed and attributed networks, which opens the door to hyperbolic manifold learning on a wide ranges of networks.

References

1. Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering*, 31(5):833–852, 2018.
2. Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *ACM SIGKDD*, pages 855–864. ACM, 2016.
3. Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguná. Hyperbolic geometry of complex networks. *Physical Review E*, 82(3):036106, 2010.

4. Ginestra Bianconi and Christoph Rahmede. Emergent hyperbolic network geometry. *Scientific Reports*, 7:41974, 2017.
5. Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1024–1034, 2017.
6. Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. Hyperbolic graph convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 4869–4880, 2019.
7. Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems*, pages 6338–6347, 2017.
8. Rik Sarkar. Low distortion delaunay embedding of trees in hyperbolic plane. In *International Symposium on Graph Drawing*, pages 355–366. Springer, 2011.
9. Christopher De Sa, Albert Gu, Christopher Ré, and Frederic Sala. Representation tradeoffs for hyperbolic embeddings. *Proceedings of machine learning research*, 80:4460, 2018.
10. Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Chang. Network representation learning with rich text information. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.
11. Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *ACM SIGKDD*, pages 701–710. ACM, 2014.
12. Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852, 2016.
13. Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv:1609.02907*, 2016.
14. James W Cannon, William J Floyd, Richard Kenyon, Walter R Parry, et al. Hyperbolic geometry. *Flavors of geometry*, 31:59–115, 1997.
15. Benjamin Wilson and Matthias Leimeister. Gradient descent in hyperbolic space. *arXiv:1805.08207*, 2018.
16. William F Reynolds. Hyperbolic geometry on a hyperboloid. *The American mathematical monthly*, 100(5):442–455, 1993.
17. Daixin Wang, Peng Cui, and Wenwu Zhu. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1225–1234, 2016.
18. Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.
19. Chengbin Hou, Shan He, and Ke Tang. RoSANE: Robust and scalable attributed network embedding for sparse networks. *Neurocomputing*, 2020.
20. Maximillian Nickel and Douwe Kiela. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International Conference on Machine Learning*, pages 3776–3785, 2018.
21. Aleksandar Bojchevski and Stephan Günnemann. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. In *International Conference on Learning Representations*, pages 1–13, 2018.
22. Xiao Huang, Jundong Li, and Xia Hu. Accelerated attributed network embedding. In *Proceedings of the 2017 SIAM international conference on data mining*, pages 633–641. SIAM, 2017.
23. Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. *arXiv:1805.09112*, 2018.